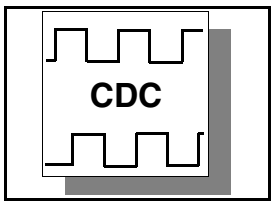
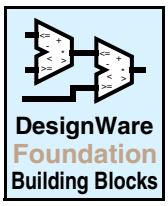




DW_sync

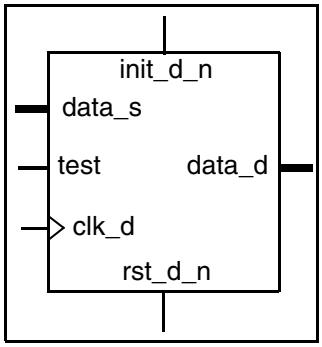
Single Clock Data Bus Synchronizer

Version, STAR, and myDesignWare Subscriptions: IP Directory

Features and Benefits

Revision History

- Parameterized data bus width
- Parameterized number of synchronizing stages
- Parameterized test feature
- Ability to model missampling of data on incoming clock domain



Description

DW_sync offers a solution for synchronizing data that crosses asynchronous clock boundaries. The DW_sync is versatile because it is configurable for the number of synchronizing stages (2, 3 or 4), the style of first-stage capturing flip-flop needed (negative or positive edge-triggered), and insertion of 'hold latches' to facilitate scan testing.

A unique built-in verification feature allows the designer to turn on a random sampling error mechanism that models skew between bits of the incoming data bus from the source domain (for more information, see "Simulation Methodology" on page 8). This facility provides an opportunity for determining system robustness during the early development phases and without having to develop special test stimulus. Figure 1-10 on page 9 and Figure 1-11 on page 10 are example timing diagrams depicting the behavior of the DW_sync when missampling is introduced.

Table 1-1 Pin Description

Pin Name	Width	Direction	Function
data_s	width	Input	Source Domain data vector
clk_d	1	Input	Destination Domain clock source
rst_d_n	1	Input	Destination Domain asynchronous reset (active low)
init_d_n	1	Input	Destination Domain synchronous reset (active low)
test	1	Input	Scan test mode select
data_d	width	Output	Destination Domain data vector

Table 1-2 Parameter Description

Parameter	Values	Description
width	1 to 1024 Default: 8	Vector width of input `data_s` and output `data_d`

Table 1-2 Parameter Description (Continued)

Parameter	Values	Description
f_sync_type	0 to 4 Default: 2	Forward synchronization type Defines type and number of synchronizing stages: ■ 0: Single clock design, no synchronizing stages implemented ■ 1: 2-stage synchronization with first stage negative-edge capturing and second stage positive-edge capturing ■ 2: 2-stage synchronization with both stages positive-edge capturing ■ 3: 3-stage synchronization with all stages positive-edge capturing ■ 4: 4-stage synchronization with all stages positive-edge capturing
tst_mode	0 to 2 Default: 0	Test mode ■ 0: No 'latch' is inserted for scan testing ■ 1: Insert negative-edge flip-flip on data_s input vector when `test` input is asserted ■ 2: For compatibility when latching is done outside of DW_sync (functions as with *tst_mode* = 0)
verif_en	0 to 4 Default: 1	Verification Enable Control ■ 0: No sampling errors inserted ■ 1: Sampling errors are randomly inserted with 0 or up to 1 destination clock cycle delay ■ 2: Sampling errors are randomly inserted with 0, 0.5, 1, or 1.5 destination clock cycle delays ■ 3: Sampling errors are randomly inserted with 0, 1, 2, or 3 destination clock cycle delays ■ 4: Sampling errors randomly inserted with 0 or up to 0.5 destination clock cycle delays For more information about *verif_en*, see “Simulation Methodology” on page 8.

Table 1-3 Synthesis Implementations

Implementation Name	Function	License Feature Required
rtl	Synthesis model	DesignWare

Table 1-4 Simulation Models

Model	Function
DW03.DW_SYNC_CFG_SIM	Design unit name for VHDL simulation
DW03.DW_SYNC_CFG_SIM_MS	Design unit name for VHDL simulation with mis-sampling enabled.
dw/dw03/src/DW_sync_sim.vhd	VHDL simulation model source code (modeling RTL) with no missampling
dw/dw03/src/DW_sync_sim_ms.vhd	VHDL simulation model source code with missampling

Table 1-4 Simulation Models (Continued)

Model	Function
dw/sim_ver/DW_sync.v	Verilog simulation model source code

Block Diagrams

Synchronization Type (*f_sync_type*) Diagrams

The following diagrams describe the behavior of the *f_sync_type* parameter.

Figure 1-1 Synchronization Type 1

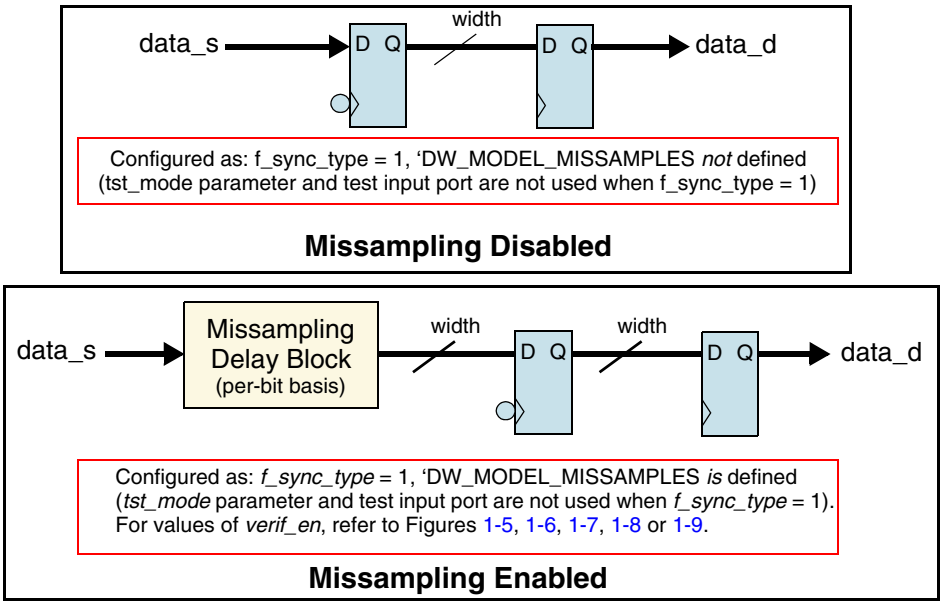


Figure 1-2 Synchronization Type 2

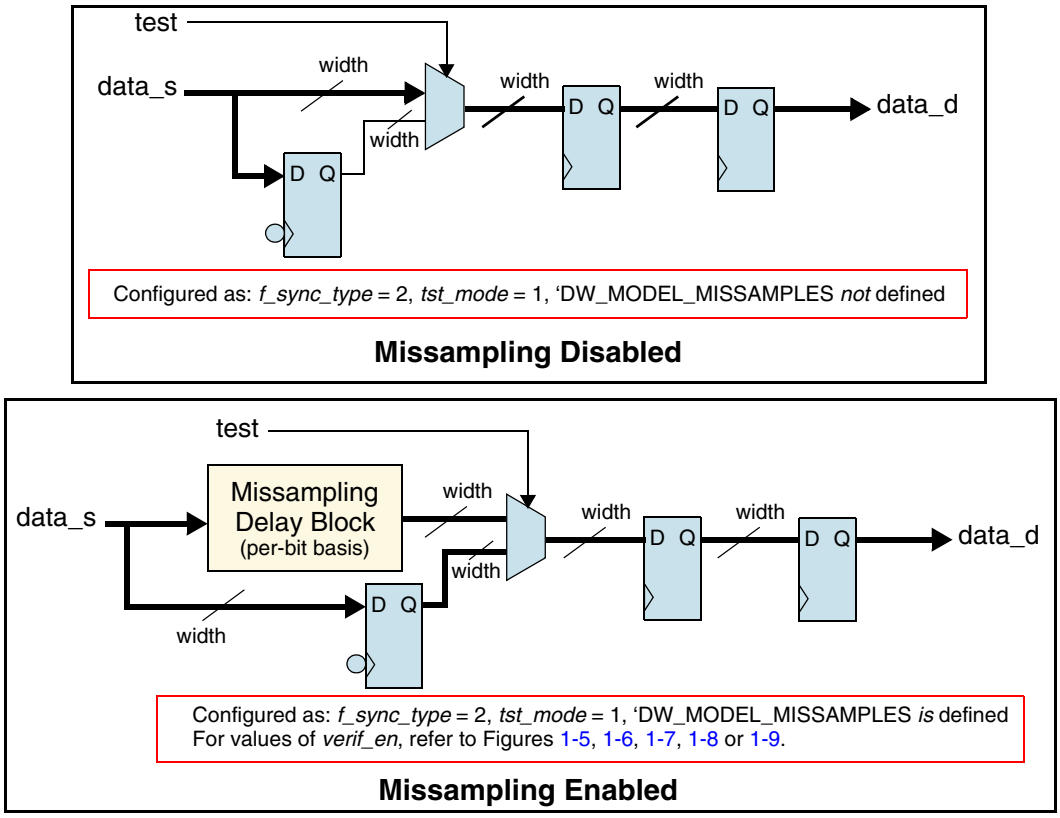


Figure 1-3 Synchronization Type 3

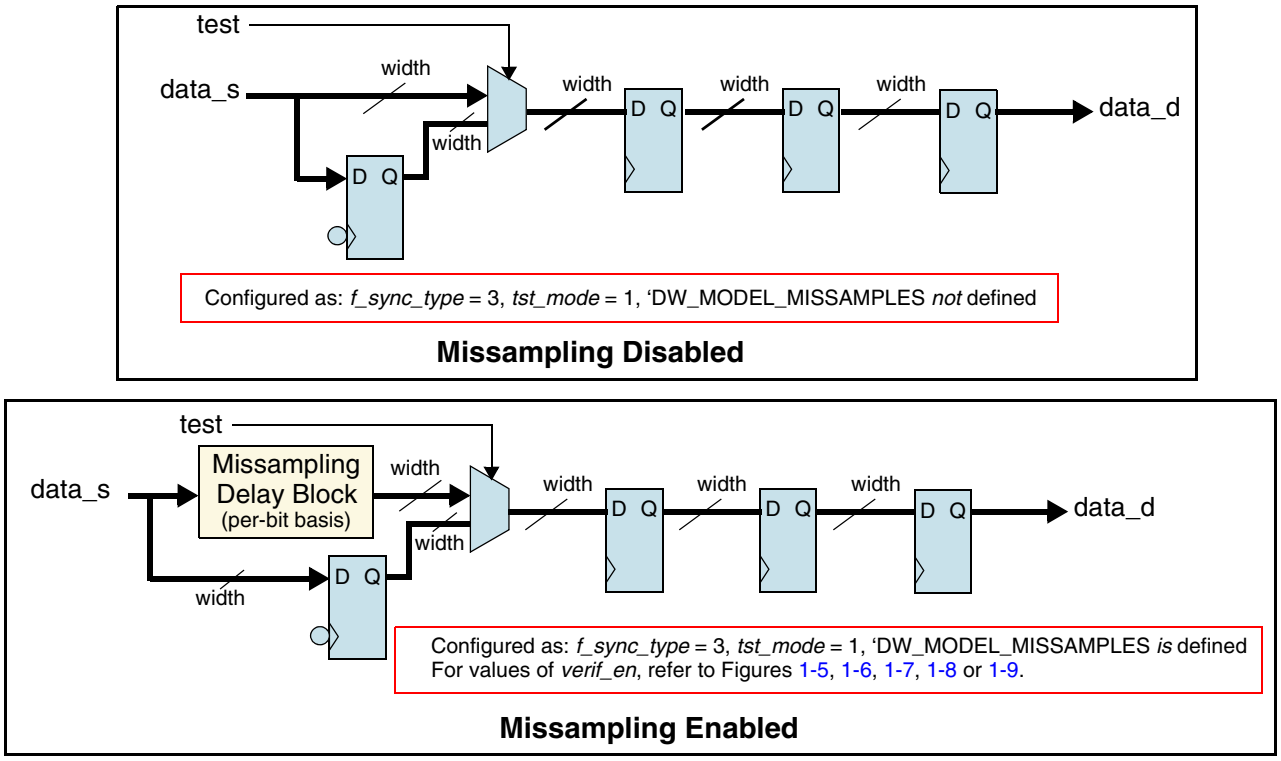
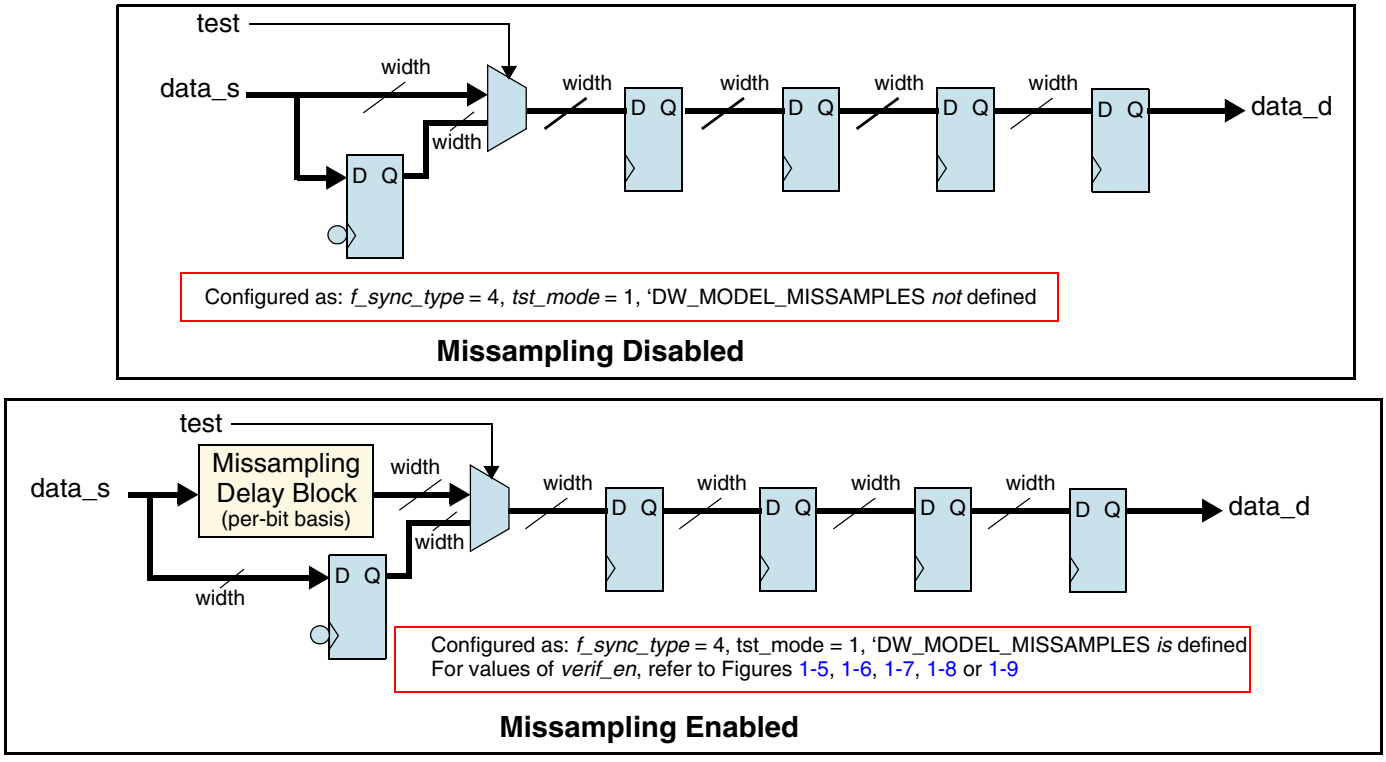


Figure 1-4 Synchronization Type 4



Verification Type (*verify_en*) Diagrams

The following diagrams describe the behavior of the *verify_en* parameter.

Figure 1-5 Missampling Model Type 0

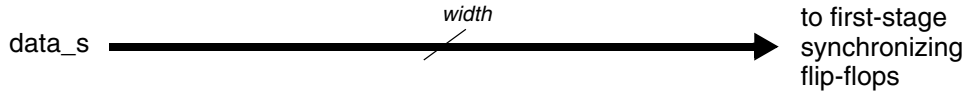


Figure 1-6 Missampling Model Type 1

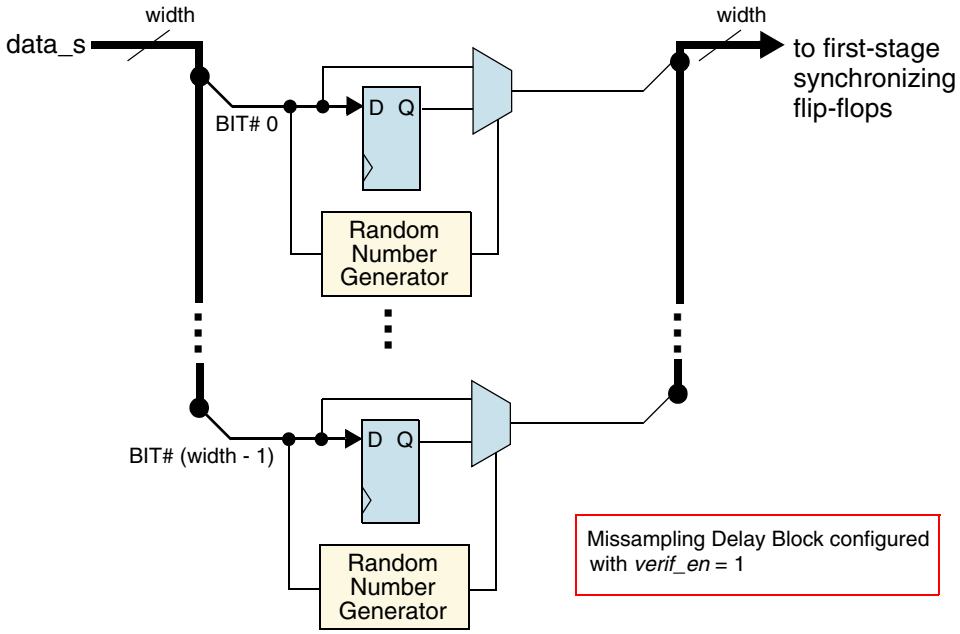


Figure 1-7 Missampling Model Type 2

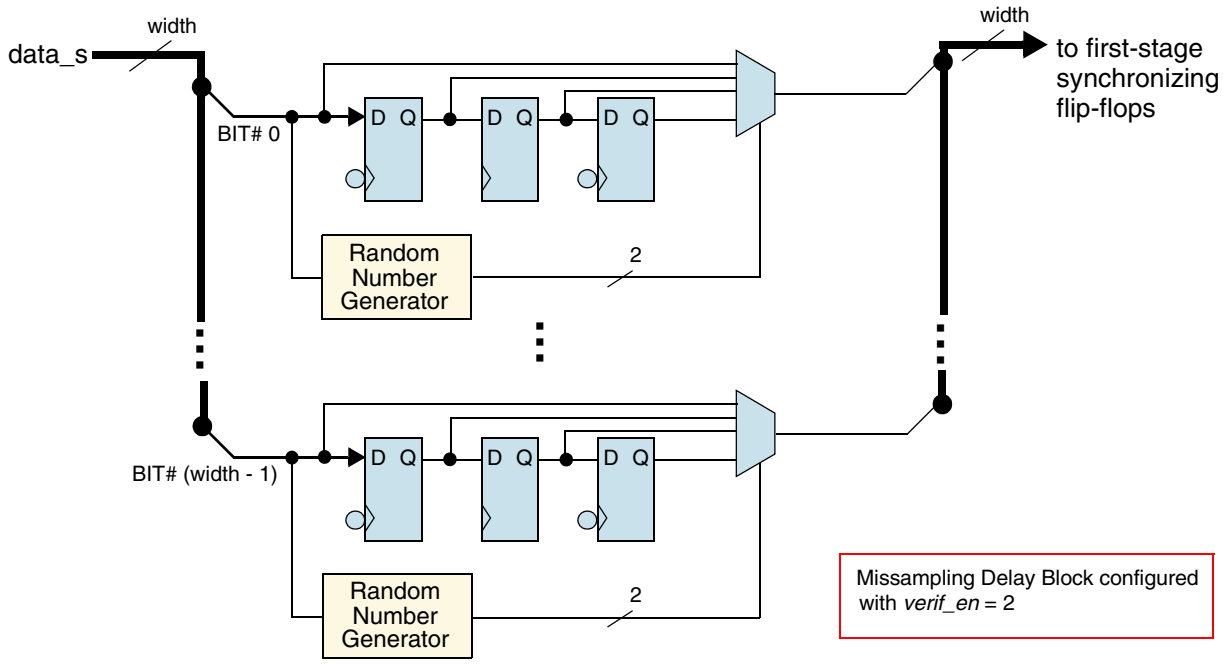


Figure 1-8 Missampling Model Type 3

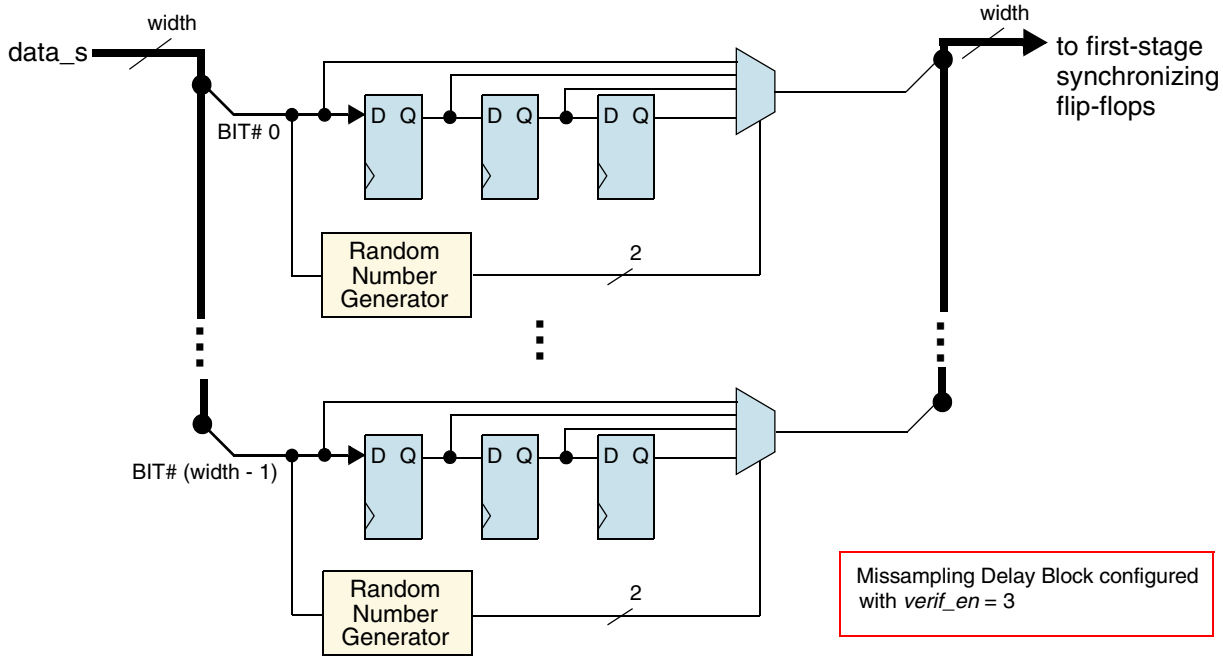
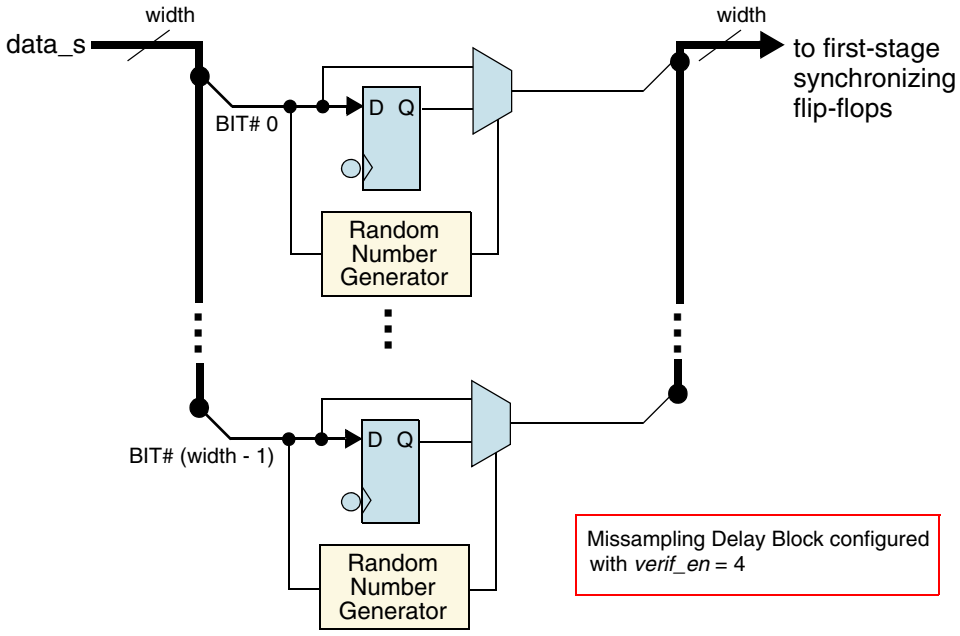


Figure 1-9 Missampling Model Type 4



Simulation Methodology

Because this component contains synchronizing devices, there are two methods available for simulation. One method is to use the simulation models that emulate the RTL model. Or, you can enable modeling of random skew between bits of signals traversing to and from each domain (called missampling).

To use the simulation models that emulate the RTL model, no special configuration is required.

To use missampling requires the following considerations:

- To enable missampling in Verilog simulations, define the macro DW_MODEL_MISSAMPLES:

  ```
  `define DW_MODEL_MISSAMPLES
  ```

  If `DW_MODEL_MISSAMPLES is defined, the *verif_en* parameter comes into play to configure the simulation model as described by Table 1-2. If `DW_MODEL_MISSAMPLES is not defined, the Verilog simulation model behaves as if *verif_en* is set to 0.

- To enable missampling in VHDL simulations, a simulation architecture named sim_ms is provided. The parameter *verif_en* only has meaning when using sim_ms. That is, when the `sim` simulation architecture is used instead, the model behaves as though *verif_en* is set to 0. For examples of how each architecture is used, see "HDL Usage Through Component Instantiation - VHDL" on page 11.

Suppressing Warning Messages During Verilog Simulation

The Verilog simulation model includes macros that allow you to suppress warning messages during simulation.

To suppress all warning messages for all DWBB components, define the DW_SUPPRESS_WARN macro in either of the following ways:

- Specify the Verilog preprocessing macro in Verilog code:

  ```
  `define DW_SUPPRESS_WARN
  ```

- Or, include a command line option to the simulator, such as:

  `+define+DW_SUPPRESS_WARN` (which is used for the Synopsys VCS simulator)

The warning messages for this model include the following:

- If values other than 1 or 0 are present on a clock port, the following message is displayed:

  ```
  WARNING: <instance_path>.<clock_name>_monitor:
      at time = <timestamp>, Detected unknown value, x, on <clock_name> input.
  ```

  To suppress only this warning message for all DWBB components, use the following macro:

  - Define the DW_DISABLE_CLK_MONITOR macro. You can define this macro in the following ways:
    - Specify the Verilog preprocessing macro in Verilog code:

      ```
      `define DW_DISABLE_CLK_MONITOR
      ```

    - Or, include a command line option to the simulator, such as:

      `+define+DW_DISABLE_CLK_MONITOR` (which is used for the Synopsys VCS simulator)

  This message is also suppressed using the DW_SUPPRESS_WARN macro explained earlier.

Timing Diagrams

Figure 1-10 Enable Missampling of data_s input

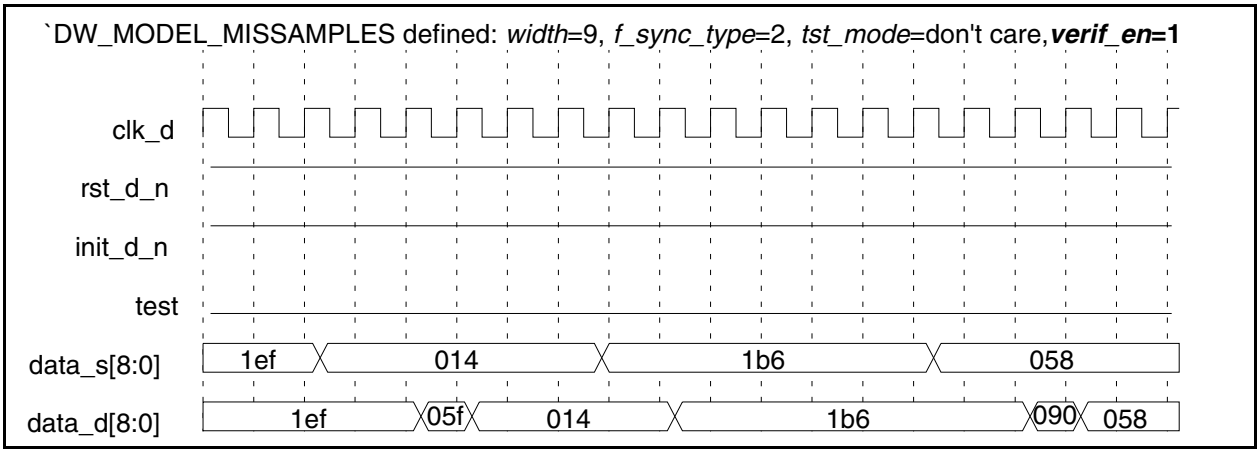
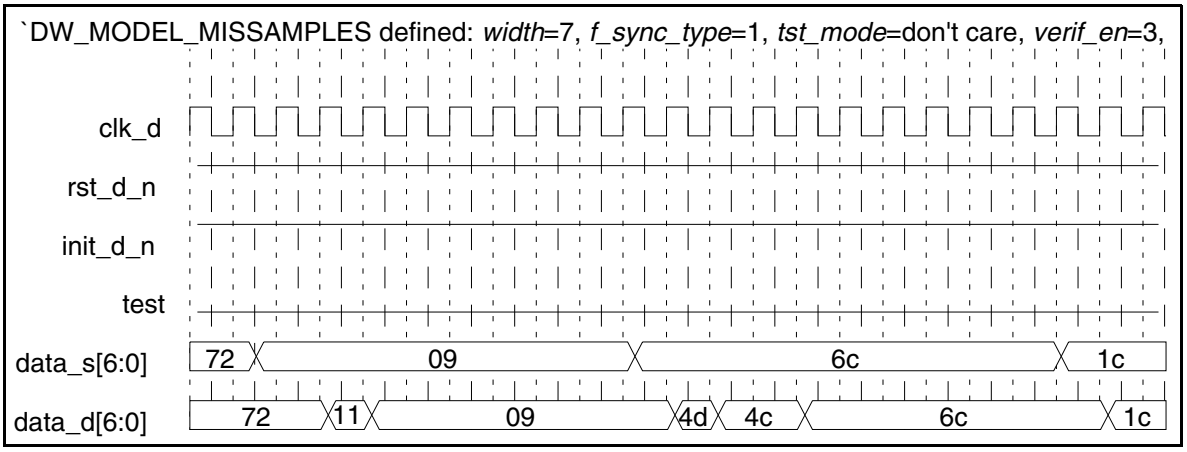


Figure 1-11 Enable Missampling of data_s input



Related Topics

- Memory – Registers Overview
- DesignWare Building Block IP User Guide

HDL Usage Through Component Instantiation - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DW_Foundation_comp.all;

entity DW_sync_inst is
      generic (
            inst_width : INTEGER := 8;
            inst_f_sync_type : INTEGER := 2;
            inst_tst_mode : INTEGER := 0;
            inst_verif_en : INTEGER := 1
            );
      port (
            inst_clk_d : in std_logic;
            inst_rst_d_n : in std_logic;
            inst_init_d_n : in std_logic;
            inst_data_s : in std_logic_vector (inst_width-1 downto 0);
            inst_test : in std_logic;
            data_d_inst : out std_logic_vector (inst_width-1 downto 0)
            );
    end DW_sync_inst;

architecture inst of DW_sync_inst is
begin

    -- Instance of DW_sync
    U1 : DW_sync
        generic map ( width => inst_width, f_sync_type => inst_f_sync_type,
    tst_mode => inst_tst_mode, verif_en => inst_verif_en )
    port map ( clk_d => inst_clk_d, rst_d_n => inst_rst_d_n,
          init_d_n => inst_init_d_n,
          data_s => inst_data_s, test => inst_test, data_d => data_d_inst );
end inst;

-- Configuration for use with a VHDL simulator
-- pragma translate_off
library DW03;
configuration DW_sync_inst_cfg_inst of DW_sync_inst is
  for inst
    -- NOTE: If desiring to model missampling, uncomment the following
    -- line.  Doing so, however, will cause inconsequential errors
    -- when analyzing or reading this configuration before synthesis.
    -- for U1 : DW_sync use configuration DW03.DW_sync_cfg_sim_ms; end for;
  end for; -- inst
end DW_sync_inst_cfg_inst;
-- pragma translate_on
```

HDL Usage Through Component Instantiation - Verilog

```
module DW_sync_inst( inst_clk_d, inst_rst_d_n, inst_init_d_n, inst_data_s, inst_test,
data_d_inst );

parameter width = 8;
parameter f_sync_type = 2;
parameter tst_mode = 0;
parameter verif_en = 1;


input inst_clk_d;
input inst_rst_d_n;
input inst_init_d_n;
input [width-1 : 0] inst_data_s;
input inst_test;
output [width-1 : 0] data_d_inst;

    // Instance of DW_sync
    DW_sync #(width, f_sync_type, tst_mode, verif_en)
          U1 ( .clk_d(inst_clk_d), .rst_d_n(inst_rst_d_n), .init_d_n(inst_init_d_n),
.data_s(inst_data_s), .test(inst_test), .data_d(data_d_inst) );

endmodule
```

Revision History

For notes about this release, see the *DesignWare Building Block IP Release Notes*.

For lists of both known and fixed issues for this component, refer to the STAR report.

For a version of this datasheet with visible change bars, click here.

Date	Release	Updates
July 2020	DWBB_201912.5	■ Adjusted the material in "Simulation Methodology" on page 8 ■ Adjusted content and title of "Suppressing Warning Messages During Verilog Simulation" on page 9 and added the DW_SUPPRESS_WARN macro
October 2019	DWBB_201903.5	■ Added the "Disabling Clock Monitor Messages" section ■ Added this Revision History table and the document links on this page

Copyright Notice and Proprietary Information

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.

All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
www.synopsys.com